THALES

PROJECT WA063A0
RDG DARWIN
REAL TIME TRAIN INFORMATION

OUTPUT PORTS

Push Ports

Socket Communications Interface Specification

WA063A01623 Issue 1 10th August 2016

Originator's signature & date	Approver's signature and date
_____	_____
D Barnard	T Parkinson

Thales, Ashurst Drive, Bird Hall Lane, Cheadle Heath, Stockport, Cheshire SK3 0XB, UK
Tel: +44 (0)161 491 4001•Fax: +44 (0)161 741 3704•ukinfo@thalesgroup.com•www.thalesgroup.com

© 2017 Thales Transport & Security Ltd - This document is the copyright of Thales Transport & Security Ltd and is issued in confidence. It must not be used other than for the purposes of the contract to which it relates and is not to be reproduced in whole or in part without the prior written permission of Thales Transport & Security Ltd.

DISTRIBUTION

1 Project Master File
2 RDG

ISSUE RECORD

Issue	Date	Purpose
1	10/08/16	Split the Push Port Interface Specification to separate the data specification from the communications specification.

CONTENTS

1. INTRODUCTION 1

2. REFERENCES 1

3. COMMUNICATION 2

3.1 Connection 2
3.2 FTP 2
3.3 Availability 2

4. PROTOCOL 3

4.1 Multiple Version Support 3
4.2 Setup Phase 4
4.2.1 Setup Schema Extensibility 6
4.2.2 Filtered Push Ports 7
4.2.3 Request TD Filtering 8
4.3 Data Phase 8
4.3.1 Darwin Status Messages 9
4.3.2 Client Requests 10
4.3.3 Updates and Snapshot Data 12

5. EXAMPLE CLIENT REQUESTS AND DARWIN RESPONSES 13

5.1 Client Requests 13
5.1.1 Setup Phase Requests 13
5.1.2 Data Phase Requests 14
5.2 Darwin Responses 16
5.2.1 Darwin Data Response Common Parts 16
5.2.2 Timetable Identifier 16
5.2.3 Snapshots 17

6. TIMETABLE FILES 18

6.1 Timetable IDs 18

7. XML SCHEMAS 19

7.1 XSD Files 19
7.2 Schema Versioning 19

8. ABBREVIATIONS AND GLOSSARY 20

OUTPUT PORTS
PUSH PORTS
SOCKET COMMUNICATIONS INTERFACE SPECIFICATION

1. Introduction

Push Ports are provided to meet the needs of external systems that deal with high volumes of enquires and that require rapid access to Darwin (RTTI) information. Darwin provides a “push” service of information in real-time that allows the client system to hold a copy of the Darwin database.

This document addresses a communications Interface Specification for the retrieval of information via the Darwin (RTTI) Push Ports Interface. The low-level communications interface to the Push Ports can vary, and the TCP socket-based interface is documented in this specification. Even though the Push Ports are accessed by different mechanisms, the data received by a client is the same and is documented in the Push Ports Data Interface Specification (Reference 2.).

Darwin makes available via the Push Port creation of, and changes to, train schedule records, together with train running predictions made by Darwin. Note that predictions and changes to schedules are relative to the original schedule as created in Darwin, usually sourced from ITPS. In order to correctly interpret Darwin data, clients must also have access to the ITPS schedule data.

Darwin also supports the download of a complete XML-format timetable for those Clients who do not have access to their own ITPS-generated timetable.

Thales will allocate a dedicated TCP/IP port and FTP account to each Client that wishes to use the socket-based Push Port Interface.

2. References

1. Common Interface File, End User Specification, Issue 28, Nov. 2012. Issued by Network Rail.

2. P75301004 Push Port Data Interface Specification.

3. Communication

Physical network connection between the Client network and Darwin is not specified here, but will typically be either a dedicated Leased WAN connection installed and maintained by an NRE supplier, or a pair of VPN connections operating over the Internet. The difference between these two options related to Client connectivity mainly involves resilience features in the event of a failure within the Darwin System.

When use is made of the dedicated WAN connection, failures within the Darwin system are transparent (at the communications level) to the Client system (other than a brief loss of connectivity). The WAN connection will be automatically re-configured to connect to an active Data Centre.

If VPN connections are used, there must be a separate VPN to each of the Darwin Data Centres and it is the responsibility of the Client to implement any logic and network routing to connect to any active Data Centre.

Security for the communication link is provided by use of the Leased WAN, or over the Internet by VPN encryption. The Darwin Firewall will only allow connection from known IP addresses to a dedicated port. Push Port information is not deemed to be highly sensitive, so these measures are considered adequate.

3.1 Connection

Push Port communication is carried out over a persistent TCP/IP socket connection, initiated by the Client. Thales allocates each Push Port Client a dedicated TCP/IP port. Only one simultaneous connection to this port is permitted at any one time.

Client systems communicate with the Darwin System by connecting to a Darwin Push Port server. The Push Port Client application must specify the following in order to make a connection to Darwin:

- IP Address of the Darwin host.
- Port Number allocated to the Push Port Client.

3.2 FTP

FTP is used by clients to retrieve timetable data and optionally snapshot data. Each client that requires FTP access will be provided with the IP address of the FTP server, a username and password. The account will be read-only and clients cannot write or delete files.

3.3 Availability

The Push Port service is available continuously, 24x365. Data is provided as soon as it is updated, other than when Darwin is re-building its timetable, when it may be queued for a short time. Darwin typically re-builds its timetable at 02:00 each day, though this may possibly occur at other times due to operational reasons.

4. Protocol

All data sent from Darwin or a Push Port Client will be formatted as XML messages. To make it easier to write code to process these messages, each individual XML message shall be preceded by an STX character (ASCII code 0x02) and followed by an ETX character (ASCII code 0x03).

To reiterate this:

All requests to Darwin *must* have a STX character preceding them and an ETX character following them.

All responses from Darwin *will* have a STX character preceding them and an ETX character following them.

A client that wishes to process a response from Darwin should read all data between an STX/ETX pair then load the message contained between these characters as an XML document.

4.1 Multiple Version Support

In order to allow new Push Port features to be introduced without forcing all existing clients to update all at once, the Darwin Push Port service supports multiple simultaneous versions. In order to provide this multiple version capability, the setup of a Push Port Client connection is split into two separate phases, a version-independent Setup Phase and a version-dependent Data Phase. Each of these phases is defined in separate XML schemas, the current versions of which can be found in section 7.1 of this document.

Figure 1 shows a flowchart of the expected actions a Client will perform when setting up a Push Port connection. The yellow blocks represent the Setup Phase and the blue blocks represent the Data Phase.

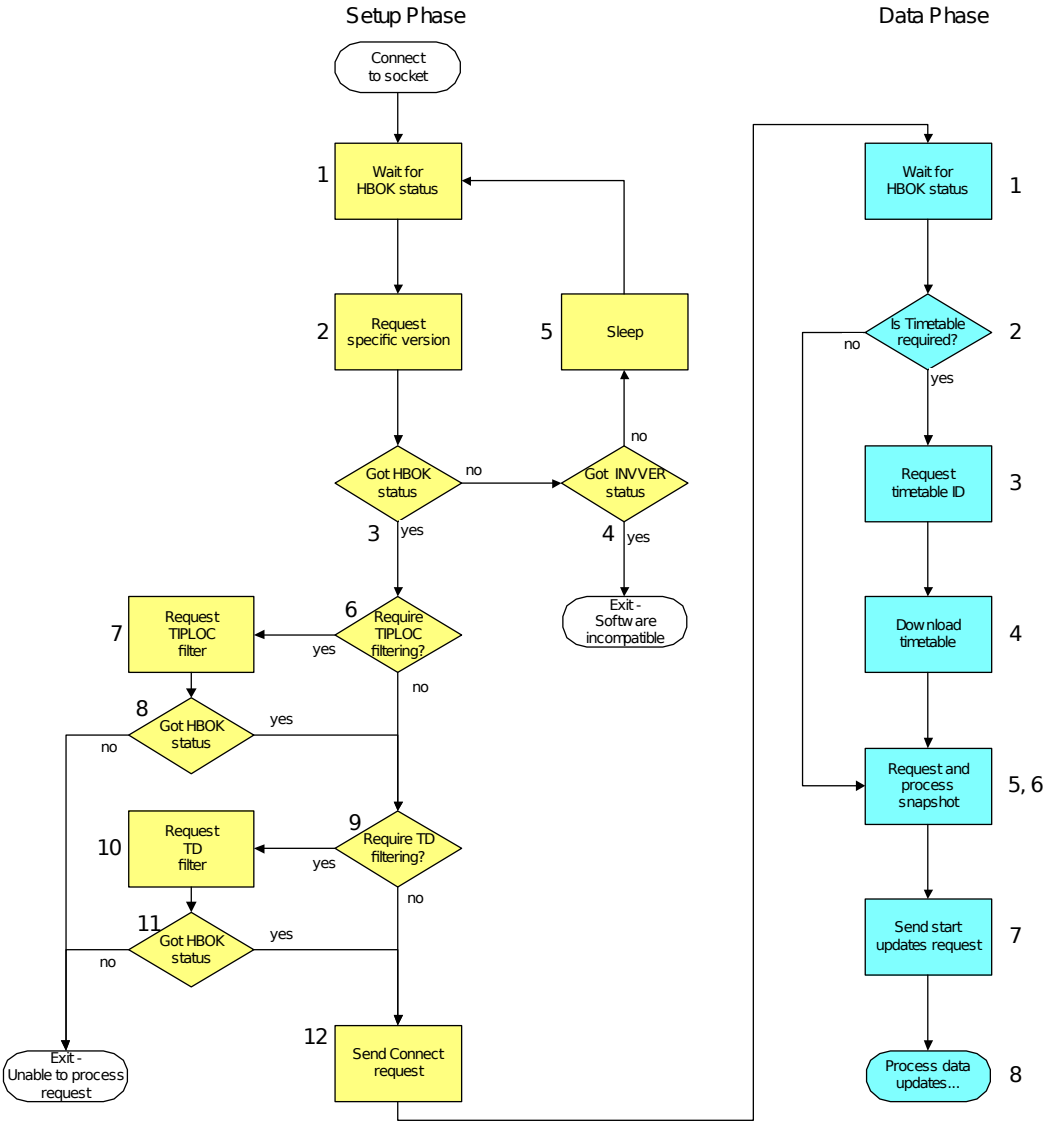

Figure 1 - Client Connection Flowchart

4.2 Setup Phase

1. As can be seen in Figure 1, after a client connects to the Push Port socket, it must wait for the server to send a *Status* message with a status of "*HBOK*" before proceeding. This allows the Push Port server to determine that it is in the correct state before accepting new connections.

```
<PPStatus code="HBOK">System is available</PPStatus>
```

2. The next step that a client is expected to perform is to request the specific versions of the data schemas that the client supports. There are three data schemas, the real-time updates schema, the timetable schema and the reference data schema. The client requests each of these individually by specifying the XML namespace of the schema. Note that if a client does not wish to download a timetable a valid ttversion and ttrefversion attribute must

still be supplied. Also note that the versions given in the examples below are not necessarily the most current or most suitable for any particular client. A client must obtain the correct values from the exact version of the schema files that they wish to use.

```
<PPReqVersion version="http://www.thalesgroup.com/rtti/PushPort/v7"
ttversion="http://www.thalesgroup.com/rtti/XmlTimetable/v5/rttiCTTSchema.xsd"
ttrefversion="http://www.thales-is.com/rtti/XmlTimetable/v1/rttiCTTReferenceSchema.xsd" />
```

3. On receipt of the version request, the server validates that it supports each of the requested schemas and returns a "*HBOK*" status if so. Each of the requested schema versions is associated with the client's connection.

4. If one or more of the requested versions is unknown then an "*INVVER*" status is returned. On receiving this status, a client is informed that it is incompatible with the Push Port server and should immediately exit.

5. If a client receives any other status that it cannot otherwise handle, it should sleep for a period (to stop busy looping) and try to make the connection again later.

```
<PPStatus code="INVVER">Invalid version requested</PPStatus>
```

6. If a client requires a *filtered* push port (see sec. 4.2.2), proceed to step , otherwise continue at step 9..

7. The client now sends a filter request with a set of filter TIPLOC codes

```
<FilterTiplocs>
    <tiploc>MNCRPIC</tiploc>
    <tiploc>STKP</tiploc>
    <tiploc>GATLEY</tiploc>
</FilterTiplocs>
```

8. In order to continue, the client must receive a "*HBOK*" status response. If any other response is received, the suggested action is for the client to close the socket and exit. If the client receives an "*INVREQ*" response to a filter request, this is a sign of a configuration error (most likely that one or more of the requested TIPLOC codes is not recognised by the Push Port server).

9. If a client requires a *TD filtered* push port (see sec. 4.2.3), proceed to step 10., otherwise continue at step 12..

10. The client now sends a TD filter request with a set of filter TD area codes

```
<RequestTD>
    <td>MS</td>
    <td>E1</td>
    <td>HN</td>
    <td>MP</td>
```

```
</RequestTD>
```

11. In order to continue, the client must receive a "*HBOK*" status response. If any other response is received, the suggested action is for the client to close the socket and exit.

12. Finally, the client will send a *Connect* message, which instructs the server to switch to the requested real-time update Data schema and operate as specified later in the Data Phase. Note that the server will not send a "*HBOK*" status response to a *Connect* message in the Setup Phase schema. If the connection is successful, the next message will be a status message in the requested Data schema.

```
<PPConnect />
```

There is the possibility that a client sends a *Connect* request before sending a *Version* request (or a client may ignore an "*INVVER*" response and try to *Connect* anyway). In this case, the server will respond with an "*INVREQ*" status then close the socket connection, thus indicating to the client that the correct protocol has not been followed.

4.2.1 Setup Schema Extensibility

The Setup schema in Figure 2 is designed to be extensible, so that different versions of clients can use the schema without breaking if the schema is updated. To achieve this, the schema is defined with just two simple element definitions, for request (incoming to Push Port server) messages and response (outgoing to clients) messages. These elements are declared with the known set of messages that can be sent as a request or response, but each also includes an <xs:any/> declaration, to indicate that any other element (from any other namespace) can also be present.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://thalesgroup.com/RTTI/PushPortSetup/root_1"
xmlns:pp1="http://thalesgroup.com/RTTI/PushPortStatus/root_1"
xmlns:pp2="http://thalesgroup.com/RTTI/PushPortFilter/root_1"
xmlns:pp3="http://thalesgroup.com/RTTI/PushPortRequestTD/root_1"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://thalesgroup.com/RTTI/PushPortSetup/root_1" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="2">
    <xs:import namespace="http://thalesgroup.com/RTTI/PushPortStatus/root_1"
schemaLocation="rttiPPTStatusV1.xsd"/>
    <xs:import namespace="http://thalesgroup.com/RTTI/PushPortFilter/root_1"
schemaLocation="rttiPPTFilterV1.xsd"/>
    <xs:import namespace="http://thalesgroup.com/RTTI/PushPortRequestTD/root_1"
schemaLocation="rttiPPTRequestTDV1.xsd"/>
    <xs:element name="PPSetupReq">
        <xs:annotation>
            <xs:documentation>Definition of request messages from clients</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice>
                <xs:element ref="pp1:PPReqVersion"/>
                <xs:element ref="pp1:PPConnect"/>
```

```
                <xs:element ref="pp2:FilterTiplocs"/>
                <xs:element ref="pp3:RequestTD"/>
                <xs:any processContents="lax"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>
    <xs:element name="PPSetupResp">
        <xs:annotation>
            <xs:documentation>Definition of response messages to clients</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice>
                <xs:element ref="pp1:PPStatus"/>
                <xs:any processContents="lax"/>
            </xs:choice>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figure 2 – rttiPPTSetupV3.xsd Schema

A client is required to **ignore** any response elements that it receives that are not defined in the version of the schema that it uses. The server will always have the most recent version of the schema (by definition) but similarly will ignore unrecognised request elements.

When the Setup schema is required to be extended, new request or response messages will be added by importing a new schema definition containing the message definitions and including the appropriate element references in the <PPSetupReq> or <PPSetupResp> elements. Note that the schema targetNamespace will *not* be updated in this case, though the version attribute may be.

The only exception to the above rule is for the <PPStatus> element, which is required to be understood by all versions of clients. To allow the capability that this element can be updated, it has been defined with the <xs:anyAttribute /> declaration. Clients are required to **ignore** any attributes of this element that they do not recognise, thus allowing future extensions to be added.

4.2.2 Filtered Push Ports

Beginning with version 2 of the Setup schema, a client may optionally request a *filtered* push port. Requesting a *filtered* port affects just the Setup phase and may be used with any supported Data phase schema version.

To request filtering, a client issues a <FilterTiplocs> request, after requesting the required schema versions and before making a <PPConnect /> request. If a <FilterTiplocs> request is not made before a <PPConnect /> request then the push port is not filtered.

Supplied with the <FilterTiplocs> request is a list of one or more <tiploc> elements. These elements define the stations for which filtered data is required. If a station has more than one TIPLOC code associated with it, then all TIPLOC values must be supplied in order for all data to be received. TIPLOC codes for multiple stations may be supplied, if required.

Normally, a client may only filter on calling points (i.e. locations with a passenger activity). Passing services, where a train does not stop at a filtered location, will not be returned. However, it is possible for Thales to enable configuration to allow passing trains to be sent. If a client requires such configuration, then a request should be submitted to NRE.

When a port is filtered, the server will only send data to a client that is relevant to one or more of the filter locations, or has been explicitly activated by the client associated with the port using the DCIS Web Service interface. Thus, updates for a service will be sent if that service calls at one of the filter locations (or is associated with a service that calls at a filter location). Station messages (<OW> elements) will only be sent if they apply to a filter location. Note that this applies equally to both “normal” update messages and snapshot messages. However, also note that the download-able Timetable files (see section 6.) will not be filtered and will always contain information for all services and locations.

Once a filtered port sends data for a service, it will continue to send updates to that service even if the service no longer meets the filter criteria. Thus, if a service is edited to call at a filter location, it will be sent to the client. If that service is subsequently edited to remove that filter location, the updates to the service will still be sent to the client. Once a service is associated with a filtered port, it will not be un-associated until the service is removed from the Darwin timetable.

4.2.3 Request TD Filtering

Beginning with version 3 of the Setup schema, a client may optionally request to receive certain data related to Train Describer (TD) information. The exact data that is received is documented in the Push Port Data Specification (reference 2.). However, in order to receive any of this data, a client must explicitly request those TD area codes for which they require data.

To request filtering, a client issues a <RequestTD> request, after requesting the required schema versions and before making a <PPConnect /> request. If a <RequestTD> request is not made before a <PPConnect /> request then no TD-related data will be sent. The order in which <FilterTiplocs> and <RequestTD> requests are made is not significant, and either, none or both may be sent in any setup request.

A <RequestTD> request includes one or more <td> elements, which contain a single two-character TD area identifier. Darwin will not validate that a particular area identifier is valid in any way, but will only send TD data that exactly matches the supplied value.

4.3 Data Phase

> Note that from version 11, the data schema has been broken into several separate files. This has been done for ease of management and to localise the impact of future changes. From a usage perspective little has changed, as there is a single root schema that includes the other files.

There are slight variations on the protocol dependent on whether the client uses train schedules sourced from Darwin and whether the client requests snapshot data via ftp or back over the Push Ports connection.

As indicated in Figure 1, Clients must follow these steps when communicating with Push Ports Server in the Data Phase:

1. The client waits for an HBOK status message.
2. If the client does not require the Darwin sourced timetable, go to step 5..
3. The client requests the timetable ID.
4. The client downloads the timetable from the Push Ports FTP server if the ID is different to the timetable the client already has, using the supplied file names.
5. The client requests a snapshot; either back over the Push Ports connection or made available on the FTP server (see section 5.1.2.3).
6. The client processes the snapshot.
7. The client sends a start update request (see section 5.1.2.4). Push Ports will start holding updates for the client from the moment the snapshot request is received. These buffered updates are sent as soon as the client sends the start update request.
8. The client processes updates until a status message with either HBFAIL or HBINIT is received, the client requests the cessation of updates or the TCP/IP connection fails.
9. If the TCP/IP connection has not failed, go to step 1..

Darwin will respond to Client requests with either the information requested or a Darwin status message to indicate the reason for the failure to meet the request. Darwin will also broadcast heartbeat messages.

A list of heartbeat and status messages is provided in section 4.3.1.

Note: A database rebuild occurs on a nightly basis. Currently, this is configured to occur at 2:00 am, although this is subject to change and should not be relied upon. During the rebuild, Darwin will not issue any update messages and the heartbeat will indicate that the database is being re-initialised (HBINIT). Once the database rebuild has completed, Darwin will notify the Client that the database is available via the heartbeat message (HBOK). At this point the client should resume the above sequence at Step 1..

4.3.1 Darwin Status Messages

Status messages are sent in response to client requests. A status message with the current state of the interface (HBOK, HBPENDING, HBINIT or HBFAIL) is also sent every 60 seconds if no other data is sent to the client in that time. This message is also sent immediately if the state of the interface has changed.

The status messages that may be received by the client are:

Code	Type	Text	Description
HBOK	Heartbeat; sent periodically.	System is available	Darwin is running and able to accept requests for data.
HBINIT	Heartbeat; sent periodically.	System is initialising	Darwin is running but is initialising its timetable. Clients should wait until a HBOK message is received.
HBFAIL	Heartbeat; sent periodically.	System is unavailable	Darwin is shutdown (the push port handler is a separate process from the core Darwin process).
HBPENDING	Heartbeat; sent periodically.	System is failing over and data is delayed	Darwin is operating, but part of the system is currently in failover mode. Data may be queued for a short period. Clients may remain connected and data will be delivered when available. This status is only returned in data schema version 11 and later.
XMLBUSY	Error; a problem was detected.	System is responding to a previous request	Darwin is busy responding to a previous request by the Client. Client should wait for the 'HBOK' code.
XMLBADIN	Error; a problem was detected.	Failed to validate request	The request XML did not match the schema.
SOFTERROR	Error; a problem was detected.	Something went wrong processing the request	An error occurred when processing the request. These errors should be reported to Thales for investigation.
XMLBADOUT	Error; a problem was detected.	Failed to generate a valid response.	Push Ports generated a response that was invalid against the schema. These errors should be reported to Thales for investigation.
TOOLONG	Error; a problem was detected.	Too much data received	Too much data has been received without a valid Push Port XML end tag.
INVREQ	Error; a problem was detected.	Dependent on circumstance.	An invalid request was made.
INVVER	Error; a problem was detected	An unsupported schema version has been requested.	A request version message was sent with schema versions not supported on the push port server.
QOVERFLOW	Error; the output queue has overflowed	Output queue overflow	The client is taking too long to process messages and the Darwin output queue has overflowed. Data will have been lost.

The general format of Darwin Data Phase status messages is as follows:

```
<?xml version="1.0"?>
<Pport  …>
  <FailureResp code="HBOK">
    System is available
  </FailureResp>
</Pport>
```

From version 11 of the data schema, DCIS clients can request (via the DCIS web service interface) a heartbeat operation, to verify full end-to-end operation. When

responding to a heartbeat request, a <FailureResp> message will include optional "requestSource" and "requestID" attributes. These attributes allow a client to detect that this heartbeat message was generated as the result of the DCIS web service request made by that client, verifying end-to-end operation. Only the client that requested the heartbeat, as determined by the "requestSource" attribute, will receive the message. The "requestID" attribute is an optional value provided by the DCIS client with their heartbeat request.

The status returned for a heartbeat will reflect the current state of the system, as will be returned in the next regular status message (assuming the state does not change in the meantime). During some internal Darwin failover scenarios, heartbeat messages may be lost, even though the regular status messages appear to indicate that the system is available.

4.3.2 Client Requests

A Client that has successfully negotiated to the Data Phase is able to send the following requests to the Darwin System:

- Timetable ID Query
- Request Snapshot
- Start/Stop Updates

4.3.2.1 Timetable ID Query

A client may request the current timetable ID to check if the Darwin timetable has been re-built during a period of disconnection. Clients that remain connected will be informed of timetable re-builds via a HBINIT status message, though the timetable ID should still be checked when the HBINIT turns into HBOK, since the ID may not always change for every HBINIT.

4.3.2.2 Request Snapshot

When a client is ready to start receiving real-time updates, after downloading a timetable if necessary, then a snapshot request must be made to synchronise with the current state of Darwin's data.

Snapshots summarise the information already held by Darwin. There are two types of snapshot available. As a part of a **Standard Snapshot**, Darwin provides information for all *activated* train journeys in the Darwin database (see the Push Port Data Specification, reference 2.), even if the service has not been modified.

As part of a **Full Snapshot,** Darwin provides the same information as the Standard Snapshot but for all services that are either *activated*, or have been modified from the base ITPS planned timetable in some way. Timetable services that have not been modified and are not active shall not be sent (though they are available in a Timetable file (see section 6.)).

As soon as a snapshot request is made, Darwin will start queuing subsequent updates so that they can be delivered when the snapshot is complete. This queue is of limited size, so to avoid it overflowing, the snapshot must be processed without excessive delay. If the queue does overflow, updates will be lost. In this

case, a one-off QOVERFLOW status message will be sent at the next opportunity after completion of the snapshot. Note that handling this error by disconnecting and reconnecting is unlikely to achieve anything. The next snapshot is likely to also have the same effect. Clients should read messages in a separate thread to the processing of them and queue them internally if they cannot process them immediately.

When the snapshot has been fully received and processed, real-time updates will only resume once a “Start Updates” request has been issued.

4.3.2.3 Start/Stop Updates

When a client wishes to start processing real-time updates following a snapshot, the “Start Updates” request must be made. This will flush the queue of updates that were collected while the snapshot was processed and allow further updates to be sent as they occur. However, if the client is unable to receive updates as fast as they are generated, the updates will be added to the queue mentioned above. If this state persists, eventually this queue may overflow and updates will be lost. In this case, a one-off QOVERFLOW status message will be sent at the next opportunity.

A client may optionally issue a “Stop Updates” request at any time after “Start Updates” has been requested. This will stop any further real-time updates from being sent and will cause the port to return to the state where it is expecting a snapshot request. Real-time updates will not be queued until a snapshot request has been made. Regular status messages will be issued in the normal way.

4.3.3 Updates and Snapshot Data

The data returned in a snapshot response and received after a “Start Updates” request is documented in the Push Port Data Specification (reference 2.) and the XML schemas (see section 7.1).

5. Example Client Requests and Darwin Responses

5.1 Client Requests

Darwin supports the following Setup Phase requests from clients (section 5.1.1).

- Request Schema Versions
- Request TIPLOC Filter
- Request TD Filter
- Connect

Darwin supports the following Data Phase requests from clients (section 5.1.2).

- Timetable ID Query
- Snapshot Requests
- Update Requests

5.1.1 Setup Phase Requests

Note that the PPSetupReq element is defined to allow *any* child elements. This allows for future extension of the protocol but does mean that XML validation is meaningless, as any well-formed XML child elements will be valid.

5.1.1.1 Request Versions

This is the first message a client will send after connecting to the push port server socket and indicates to the server what versions of the data, timetable and reference data schemas the client will use.

```
<?xml version="1.0" encoding="utf-8"?>
<PPSetupReq xmlns="http://thalesgroup.com/RTTI/PushPortSetup/root_1">
  <PPReqVersion xmlns="http://thalesgroup.com/RTTI/PushPortStatus/root_1"
        version="http://www.thalesgroup.com/rtti/PushPort/v9"
        ttversion="http://www.thalesgroup.com/rtti/XmlTimetable/v7/rttiCTTSchema.xsd"


ttrefversion="http://www.thales-is.com/rtti/XmlTimetable/v2/rttiCTTReferenceSchema.xsd"
/>
</PPSetupReq>
```

Note that the versions given in the example above are not necessarily the most current or most suitable for any particular client. A client must obtain the correct values from the exact version of the schema files that they wish to use.

Any attempt to send any other message before a request versions message will result in an INVREQ status message response.

Any attempt to request versions that are not supported by the push port server will result in an INVVER status message response.

5.1.1.2 Request TIPLOC Filtering

This is an optional message that a client may send to request a *filtered* push port. The message identifies the station locations for which filtered data is required.

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSetupReq xmlns="http://thalesgroup.com/RTTI/PushPortSetup/root_1"
xmlns:pp1="http://thalesgroup.com/RTTI/PushPortFilter/root_1">
   <pp1:FilterTiplocs>
      <pp1:tiploc>MNCRPIC</pp1:tiploc>
      <pp1:tiploc>STKP</pp1:tiploc>
      <pp1:tiploc>GATLEY</pp1:tiploc>
   </pp1:FilterTiplocs>
</PPSetupReq>
```

If filtering is requested, it must be acknowledged with a HBOK status message in order for a subsequent Connect request to succeed.

5.1.1.3 Request TD Filtering

This is an optional message that a client may send to request certain TD-related data from the push port. The message identifies the two character TD Area identifiers for which data is required.

```
<?xml version="1.0" encoding="UTF-8"?>
<PPSetupReq xmlns="http://thalesgroup.com/RTTI/PushPortSetup/root_1"
xmlns:pp3="http://thalesgroup.com/RTTI/PushPortRequestTD/root_1">
   <pp3:RequestTD>
      <pp3:td>MS</pp3:td>
      <pp3:td>MP</pp3:td>
   </pp3:RequestTD>
</PPSetupReq>
```

If filtering is requested, it must be acknowledged with a HBOK status message in order for a subsequent Connect request to succeed.

5.1.1.4 Connect

This indicates to the Push Port server that the setup phase is complete for this client and that this client will now start using messages defined within the requested data schema version, these requests are described in section 5.1.2.

```
<?xml version="1.0" encoding="utf-8"?>
<PPSetupReq xmlns="http://thalesgroup.com/RTTI/PushPortSetup/root_1">
  <PPConnect xmlns="http://thalesgroup.com/RTTI/PushPortStatus/root_1" />
</PPSetupReq>
```

Any attempt to connect before a valid request versions message will result in an INVREQ status message response.

5.1.2 Data Phase Requests

5.1.2.1 Push Port Data Requests Common Part

All Darwin data requests are wrapped in a **<Pport>** element. This element includes a timestamp, the version of the data schema used and the required namespace directives. An example is provided here:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<Pport ts="2010-05-21T11:00:37" version="11.0"
xmlns="http://www.thalesgroup.com/rtti/PushPort/v11">
```

The version attribute should be set to the version number of the data schema used (see section 7.1) and the ts (timestamp) should be set to your current local time. The namespace *must* match the namespace identified in the “Request Versions” setup phase request, and may not match the example given above (if a schema other than v11 is being used).

In the following examples, the common attributes in **<Pport>** elements are omitted for clarity.

5.1.2.2 Timetable ID Query

This allows the Client to query for the current timetable identifier and the names of the timetable and reference data files that are available for download. The filenames returned are dependent on the timetable and reference data versions requested by the client when it connects.

The Client can request the timetable identifier at anytime.

```
<Pport …>
   <QueryTimetable />
</Pport>
```

If a timetable rebuild is ongoing or Darwin is unavailable then Darwin will respond with a HBFAIL status message.

5.1.2.3 Snapshot Request

Allows the client system to request a snapshot of train information, as described in section 4.3.2.2. The standard snapshot request is:

```
<Pport …>
   <GetSnapshotReq />
</Pport>
```

The full snapshot request is:

```
<Pport …>
   <GetFullSnapshotReq />
</Pport>
```

Note that either of the snapshots can request that data be made available via FTP as an XML file compressed using gzip:

```
<Pport …>
   <GetSnapshotReq viaftp="true"/>
</Pport>

<Pport …>
   <GetFullSnapshotReq viaftp="true"/>
</Pport>
```

5.1.2.4 Start Update Request

This message will trigger the start of updates from Darwin. Updates include any updates to any of the journeys held in Darwin including movement reports, schedule changes, etc. See the Push Port Data Specification (reference 2.) for an overview of the data sent.

```
<Pport …>
   <StartUpdateReq/>
</Pport>
```

5.1.2.5 Stop Update Request

This message is used to stop the updates from Darwin.

```
<Pport …>
   <StopUpdateReq/>
</Pport>
```

5.2 Darwin Responses

Due to the volume of data to be transmitted, the data provided by Darwin is structured to minimise bandwidth. This has led to the use of abbreviations in the choice of tag and attribute names.

All messages are time-stamped on transmission with the current time (i.e. the time taken from Darwin's internal clock). The messages conform to the W3C XML definitions.

There is no indication of GMT/BST in the data provided. This is intentional as it is not possible to reliably establish whether the data flowing into Darwin is GMT or BST. Times should be assumed to be local time.

The following sections detail the responses from Darwin to the requests made by the Client as described in Section 5.1.

5.2.1 Darwin Data Response Common Parts

All Darwin data responses are wrapped in a **<Pport>** element. This element includes a timestamp, the version of the schema used on the server and the required namespace directive. An example is provided here:

```
<?xml version="1.0" encoding="UTF-8"?>
<Pport ts="2010-05-21T11:00:37" version="11.0"
xmlns="http://www.thalesgroup.com/rtti/PushPort/v11">
```

The version attribute will be set to the version number in the schema definition (see section 7.1) and the ts (timestamp) will be set to the local time on the Darwin system. The namespace will match the namespace identified in the "Request Versions" setup phase request, and may not match the example given above (if a schema other than v11 is being used).

In the following examples the common attributes in **<Pport>** elements are omitted for clarity.

5.2.2 Timetable Identifier

The timetable identifier and timetable/reference data filenames will be communicated via a Timetable ID element.

```
<TimeTableId ttfile="20100524033206_v7.xml.gz"
             ttreffile="20100524033206_ref_v1.xml.gz">20100524033206</TimeTableId>
```

The `ttfile` and `ttreffile` attributes identify the names of the timetable files that can be downloaded via FTP. The filenames returned are dependent on the timetable and reference data versions requested by the client when it connects.

5.2.3 Snapshots

Upon request, Darwin will provide a snapshot to a Client. Further requests from the same Client shall be queued until the snapshot completes.

Snapshots will provide the following data:

- Schedule Information (‘schedule’)
- Actual and Forecast Information (‘TS’)
- Association information (“association”)
- Table Suppression and Station Messages (‘OW’)
- Train order information (“trainOrder”)
- Train Alert Messages (“trainAlert”)
- Alarms (‘alarm’)

Snapshots use the same XML format as updates (see below). Snapshots are presented in a collated form using the <sR> element in place of the <uR> element.

Snapshots can be delivered in two forms, either back over the connection or from the Push Ports FTP server.

5.2.3.1 Snapshots over the Push Ports Connection

In this instance, the resulting XML is sent back over the TCP/IP connection to the client.

5.2.3.2 Snapshots via FTP

If the client requests that the snapshot be retrievable via FTP then the snapshot that would have been sent back is saved to a file and compressed using gzip. A message then is sent indicating the filename that a client should use to fetch the snapshot data.

```
<SnapshotId>filename<SnapshotId/>
```

No assumptions should be made on the format of the filename, though it is guaranteed to contain only alphanumeric characters and periods, and to be unique between clients connected in parallel.

Details of the FTP account provided are in section 3.2. The client has read access to the following directory:

/snapshot	This directory holds snapshot data if it was requested via FTP.

Since FTP snapshots are compressed, large snapshots may be received significantly quicker than when received over the Push Port socket connection.

Snapshot files are automatically deleted by Darwin, usually daily. A client does not need to (and cannot) delete them itself.

6. Timetable Files

Darwin makes available timetable files to registered clients. The format and contents of these files is detailed in the Push Port Data Specification (reference 2.) and the Timetable XML schema files.

6.1 Timetable IDs

A Timetable ID is used to identify the currently available timetable files. A client can obtain the current Timetable ID by calling the Timetable ID Query (see Section 5.1.2.2) to determine the timetable ID and file names (Section 5.2.2). Clients that determine that the timetable has changed since their last enquiry may then download the new files. However, note that it is not guaranteed that the Timetable ID will change after every HBINIT status.

Note that for backwards compatibility reasons, the Push Port service will not generate a HBOK status until the Timetable files have been made fully available. However, there is no longer any reason why a client must fully download and process the timetable before starting to process real-time updates by generating a snapshot and starting updates (sections 5.1.2.3 and 5.1.2.4).

Since all schedules are now “activated” before any other data is published for them, it is safe to process the timetable in parallel with processing real time updates. The only need for the timetable is to gain access to those schedules that will be activated in the future.

Note also that a new timetable will not be generated without the Push Port transitioning through a HBINIT phase. Therefore, there is no point in “polling” the Timetable ID Query (or the timetable files themselves). A check for the timetable need only be made when a HBOK is received after a HBINIT, or on a new socket connection.

7. XML Schemas

7.1 XSD Files

The XML schema XSD files are available from the same source as this document.

The latest setup schema version is:

Schema	Version	Namespace	Filename
Setup	3	http://thalesgroup.com/RTTI/PushPortSetup/root_1	rttiPPTSetup_v3.xsd

The latest Data, Timetable and Reference Data schema versions are listed in the Push Port Data Specification (reference 2.).

7.2 Schema Versioning

All the schemas belong to namespaces that include version numbers. The Push Port Service itself supports multiple versions from version 1 of the Setup Schema and version 7 of the Updates Schema onwards and the namespaces are used to indicate the version required. The namespaces are also used as a defence against clients using versions of the schema incompatible with the software.

8. Abbreviations and Glossary

ATOC	Association of Train Operating Companies (now RDG)
CIF	Common Interface File. The format of this file defines the format in which ITPS provides schedule information.
CRS	Computerised Reservation System
DCIS	Darwin CIS. Interface between Darwin and CIS systems driving passenger displays at stations.
False Destination	A train destination. Typically used in a circular route to provide a route for the train.
FTP	File Transfer Protocol.
Gzip	A compression tool using the DEFLATE format as defined in RFC 1951. gzip is defined in RFC 1952.
ITPS	Integrated Train Planning System (replacement source of schedule data for TSDB).
LDB	Live Departure Boards. This is the publicly available web interface to the Darwin system.
RDG	Rail Delivery Group
RID	Darwin generated ID. A unique ID held within the Darwin database to identify a journey.
RTTI	Real Time Train Information – database of train running information. Previous name for Darwin.
Snapshot	There are two types of snapshot: 1. Standard Snapshot: information for all train journeys in the Darwin database that are in progress or have yet to commence. 2. Full Snapshots: includes information for all journeys in the Darwin database since the last timetable rebuild. I.e. includes the standard snapshot plus historic schedule information. Snapshot data is defined in Reference 2..
Theseus	Theseus supplies train-running data received from TOC's to Darwin.
TIPLOC	Timing Point Location
TOC	Train Operating Company
TSDB	Train Services Database (now superseded by ITPS).
Updates	Darwin provides update information to the Client when this information becomes known to Darwin. Update data is defined in Reference 2..
UID	Unique Identifier. (However, the UID is not always unique within the Darwin database)